Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 83 (2016) 300 – 304

Procedia Computer Science

The 7th International Conference on Ambient Systems, Networks and Technologies (ANT 2016)

Resource Usage Analysis of a Sensor-Based Mobile Augmented Reality Application

Ahmet Karaman[a], Doga Erisik[a], Ozlem Durmaz Incel[a,*], Gulfem Isiklar Alptekin[a]

[a]*Department of Computer Engineering, Galatasaray University, Ciragan Cad. No:36, Ortakoy, Istanbul, 34349, Turkey*

Abstract

In this work, design of an Android-based augmented reality application is presented and particularly its performance is analyzed in terms of resource usage in comparison to similar applications. The application displays merchant, branch information of one of the Turkish banks, as well as related sales campaigns of the merchants on the screen that are within the proximity of the user's location. The developed application uses GPS, compass, gyroscope, accelerometer sensors and it utilizes an accurate tagging algorithm. We examine the resource and battery consumption of the application. Accordingly, we propose methods for improving the resource usage. The proposed improvements reduce the resource consumptions up to 35% and the application performs considerably well compared to the state of the art commercial applications. We believe that the suggested improvements can be useful for other sensor-based mobile augmented reality applications.

Peer-review under responsibility of the Conference Program Chairs

Keywords: Augmented reality; mobile application; mobile phone sensing; performance evaluation.

1. Introduction

Augmented reality (AR) applications help users interact with their surroundings via mobile devices. Many of these applications; using phone's GPS (Global Positioning System), compass and other sensors (gyroscope, accelerometer etc.), show information on the display identifying the user's whereabouts and provide direction/navigation information. In most of the studies in the literature that utilize sensors on smart phones, data is collected on the phone and more powerful processes in the data processing is performed off-line on a computer. When the limited processing capabilities and battery capacity of the phones are taken into consideration, such a scheme may not be efficient especially if real-time performance is required. Thus, for the development of real-time applications, resource constraints on the phones should be considered.

In spite of the increase in processing power, feature set, and sensing capabilities, the smartphones continue to suffer from battery life limitation, which hinders the active utilization of LBA's (location-based application) [1]. Unfortunately,

* Corresponding author. Tel.: +90-212-2274480
E-mail address: odincel@gsu.edu.tr

Peer-review under responsibility of the Conference Program Chairs
doi:10.1016/j.procs.2016.04.129

GPS, the core enabler of LBAs, is power-intensive, and its aggressive usage can cause a complete drain of the battery within a few hours. The LBA developers are suggested to reduce the use of GPS by increasing the location-update intervals (say, to more than a minute), thus allowing GPS hardware to sleep between successive location-updates. Such a simple solution can improve battery life by forcing applications to request location information less frequently, but it has a fundamental limitation[1]. In AR applications, location fixes must be taken more frequently. At the same time, other sensors must be working all the time.

In this work, we focus on an AR application entitled "SARAS (Sensor-based Augmented Reality Application Software)" which shows the bank merchants and branch information on the device screen. We particularly analyze the resource consumption of the application and aim to improve its performance in terms of energy and resource consumption. The application works as follows: the user gets the viewing angle (or framing) within the camera to be launched in SARAS by looking at a direction in shopping centers, on the street or on the road (highway, city). If there are points of interest (POIs) related to the bank in the viewing angle (also inside framing), this information appears as a list on the screen. If a point is selected from the list, detailed information (such as a campaign) is displayed.

The main contribution of this work is to analyze the energy and resource usage of AR applications, particularly SARAS. First of all, a detailed performance analysis was performed to find the battery and power consumption by using Android platform performance tracing tools. It was observed that keeping the resources on and calling the location-distance function for each POI were the most CPU consuming factors. Therefore, it was provided as a solution to close resources on the activity passings and to calculate location-distance on the web service call. After various improvements the application is observed to consume 35% less energy. Besides, different GPS and sensor processing, and an accurate tagging algorithm are applied to the application and then resource consumption is re-examined. Overall, resource consumption on SARAS is shown by making comparisons to several similar applications.

2. Related Work

Zhuang et al. study the energy efficiency of location sensing in[1]. First, they put forth of battery effects of location sensing, then they suggest new location sensing methods. They explain that these methods improve the battery life by up to 75% by reducing the number of GPS invocations. This work is close to our study in terms of location sensing. However, in our work other factors in AR applications are also considered and different GPS and sensor processing methods are analyzed in terms of energy consumption.

Sarmiento et al. evaluate the performance of Android systems for AR applications[2]. For image capturing, they offer to use native code processor to improve velocity of execution and they analyse the performance with different image formats and capture frequencies. For the tracking sensors (accelerometer, compass), they measure values while the user is walking and for the GPS they analyze type of network connections on Android platforms. Since they use image rendering-based AR for the testing, they mostly concentrated on image capturing and Android graphics performance. Finally, they present the experimental results and conclusions over the basic test application.

Wagner et al. also study AR on mobile phones over image rendering-based methods in[3]. They also examine the performance of rendering, memory, bandwith usage and networking with different type of image rendering programs (OpenGL and Direct 3D). Another interesting research in this area is based on cloud computing[4]. Chen et al. claim that AR over the cloud computing has a great potential and the power of cloud computing can solve the limitation problems of AR applications. Some computational tasks can be performed on the cloud with service as a server method. These two AR studies are based on the image rendering conventional applications. Whereas our solution utilizes a GPS-based solution and we analyse its performance considering many different factors. There are also commercial applications similar to the SARAS application, such as LAYAR[5], Wikitude[6]. We provide performance comparisons with LAYAR in Section 4.

3. SARAS Application

The augmentation data source is the bank merchants and the campaigns available. This data is stored in a remote database and loaded via the RESTful web services to the phone's local database. The data stored in the database includes the merchant/branch id, name, location (latitude, longitude, floor) and campaign codes. While testing the application, database included information on 296 merchants and branches. SARAS uses the front camera, current location info, motion and direction of the device for augmentation data. Doing so, device camera API's, graphics

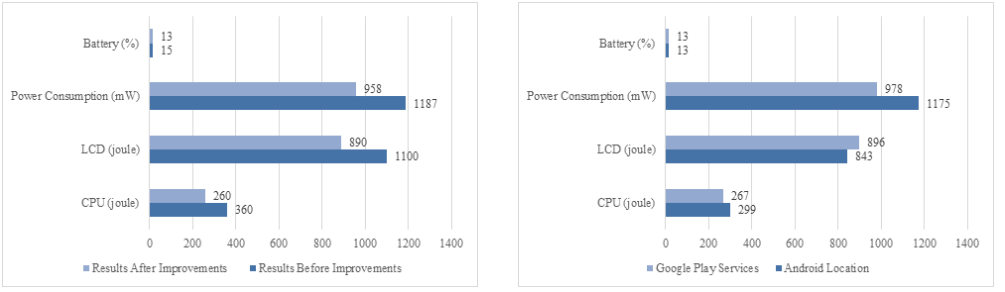
API and the motion sensors are used. They must be working continuously in the application. For an extended battery lifetime and good performance, the resources must be used efficiently by the application.

One of the main resources in an AR application is the camera. Device camera is always open in the application's main activity. The camera display is shown on the screen and the augmented data are placed on this display. In SARAS, camera display is created by using SurfaceHolder object. Another important task in AR apps is to find the user's location. The application finds nearby POIs according to this information. Location information is obtained using GPS. In SARAS, Android Location Library is used for GPS properties. Besides, an available Google Play Services API is used if the device supports. Google Play Services provide Google-powered features such as Maps, Google+, etc. Permissions need to be added to manifest file to access location libraries and Google Play Services. SARAS application supports three different GPS modes for both Google Play Service version and Android Location Library versions. Detailed test results obtained with different modes are explained in Section 4. These modes are:

1. **Normal Mode:** It is the default mode in which the application starts. The application is forced to take updates every 20 seconds. This should consume less energy compared to the drive mode. Detailed results are given in Section 4. This mode can be use while the user is walking.
2. **Drive Mode:** The application is forced to request updates every 1 seconds. This mode definitely consumes more energy compared to the normal mode. This mode can be used while the user is driving or moving quickly.
3. **No Updates Mode:** The application does not request location updates. This mode consumes less energy than other modes. It can be used while the user is stationary. It assumes that the user is at the same place and make its operations based on the last GPS information taken.

4. SARAS Performance Analysis

In this section, we analyze the performance of SARAS application in terms of resource usage, particularly battery consumption. The PowerTutor application is used for the resource usage measurements[7]. PowerTutor is an application for Android platform that displays the power consumed by major system components, such as CPU, network interface, display (LCD), and GPS receiver and different applications. The application allows software developers to see the impact of design changes on power efficiency. PowerTutor uses a power consumption model built by direct measurements during careful control of device power management states. This model generally provides power consumption estimates within 5% of actual values. As mentioned, SARAS supports two GPS methods. Google Play Services and Android Library. The application runs in different sensor perception, like normal mode or drive mode. The application is tested under these different methods and modes. HTC One Mini phone was used in the tests. In each test, the application was used for 20 minutes continuously.



(a) Performance Results after Improvements (b) Comparison of GooglePlayService & Android Location

Fig. 1. Improvements and Google Play vs Android Location Comparisons

4.1. CPU Profiling

In this section we explain the method utilized for CPU profiling at thread level and show the analysis of the profiling with the improvements added to SARAS. While the application is running, we start by selecting the "Start Method Profiling" on the running thread on DBMS. It provides results of the CPU consumptions at function level. Considering the results of the systrace, it was seen that sensors and screen placement functions consume more CPU compared to other functions. It is an expected result since these functions are always running in the application life cycle. Hence, all these functions are examined again. For example; all information in the merchant web service is fetched to a local

database at first and then, they are filtered by the parameters in the menu. The distance filter control is provided with the distance calculation for each POI by the Android GPS library. This affected the performance in a negative way. So, the SQLite query that searches the local merchant data has been improved with the distance information calculation. The bank merchant web service was returning only the GPS and name information. In the map view activity for each merchant the campaign information was taken from another web service. In order to prevent this, a Boolean flag that shows the campaign of the merchant is added to the merchant web service. All these improvements contributed to decrease CPU and LCD power consumption in runtime. The CPU consumption decreased to 266 joules from 360 joules and the LCD consumption decreased to 890 joules from 1100 joules for twenty minutes of usage. Figure 1(a) shows the results after performance improvements. These results were taken in normal mode with the Google Play Services support. The battery usage is given in percentage, the CPU and LCD are given in joules and the total power consumption is given in *mW*.

4.2. Using Google Play Service versus Android Location

SARAS uses Google Play Services library to take GPS fixes as default. However, some devices do not support this library. In such a case, SARAS uses Android Location Library. First of all, these two GPS methods are tested in terms of resource consumptions, as in Figure 1(b). As shown in Figure 1(b), using Google Play Service library consumes less CPU and power consumption. LCD power consumption values depend on the number of merchants displayed on the screen. These values are close to each other. The only exception is the battery life.

4.3. Normal Mode versus Drive Mode

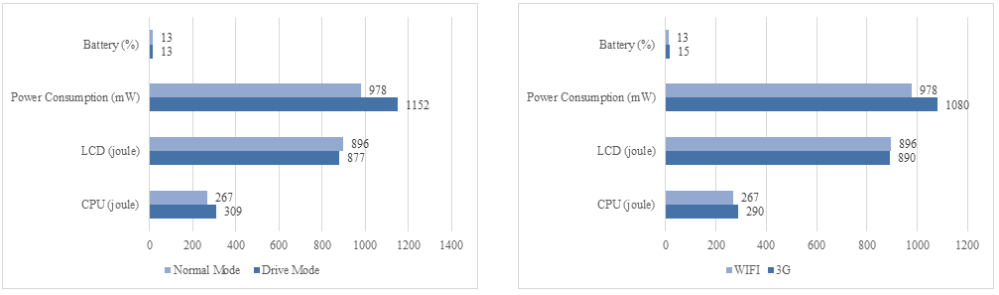
Users can use SARAS while walking or driving or with any transportation mode. GPS update intervals must be different in these different cases. SARAS takes GPS fixes 10 times faster in drive mode. Figure 2(a) shows the test results of this comparison. These results are taken with the Google Play Services support. Since more GPS sampling is performed in the drive mode, it consumes more battery.

4.4. Impact of Connection Type: 3G versus WiFi

The effect of connection type is also significant on energy consumption in AR applications. The application requires internet connection. SARAS is tested with WiFi and 3G connection types. The results show that with 3G connection, battery and CPU consumption are higher than the usage with WiFi connection as shown in Figure 2(b). The reason of this is that the device connects to a remote base station while using 3G, however using WiFi the device connects to a closer access point.

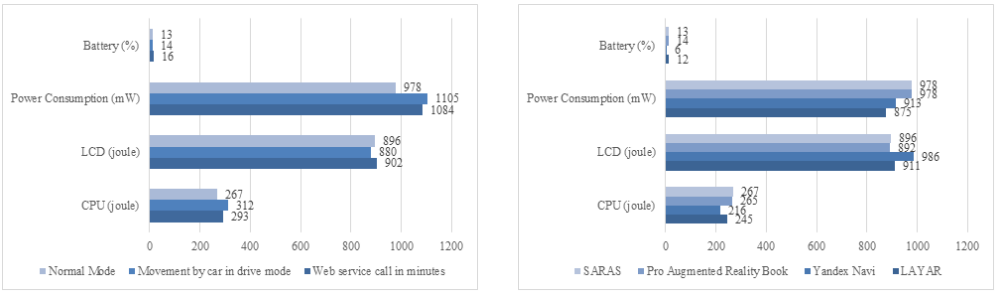
4.5. Impact of Web Service Call Frequency

SARAS calls the merchant web service when the application is started. It is not called again if the user does not change the location by at most 10*km* as the default value. In order to measure the impact of the frequency of calling web services, tests are conducted with different frequency values. For example, a test is conducted by calling the service in minutes intervals or calling at a lower location distance change in the drive mode. For 20 minutes, test merchant web service is called 20 times in the first test. In the drive mode, it was called 4-5 times depending on the location change. Figure 3(a) shows the increase of battery consumption when the web service call frequency increases. In the drive mode tests, 3G connection is used. Hence, CPU and power consumption values are higher than the other case.



(a) Comparison of Normal Mode & Drive Mode (b) Impact of Connection Type

Fig. 2. Connection Type and Web Service Call Comparisons



(a) Impact of Web Service Call Frequency (b) Comparison with Other Applications

Fig. 3. Connection Type and Web Service Call Comparisons

4.6. Comparison with Other Applications

Similar applications like LAYAR are also tested with PowerTutor. Tests are preformed with the same device and connection type. Besides, another accurate tagging AR algorithm from Pro Android Augmented Reality book[8] is developed for test purposes. When we look at the results shown in Figure 3(b), this algorithm consumes more battery, but less CPU compared to our algorithm. This is due to the fact that, the process is called more often but calculations are less than in our algorithm. SARAS performs similarly when compared with similar applications. In Yandex Navi application, route calculation/navigation functionality is not used. Hence, it consumes less battery. SARAS consumes only 58 joules of CPU and 9% battery.

To summarize and generalize our findings for other mobile AR applications, we can say that CPU profiling should be performed for identifying resource-intensive methods used in the development. The two popular location libraries, Android Location and Google Play Service exhibited similar behaviours in terms of battery consumption and hence both can be used. The web service call frequency impacts the results. Hence, it would be more efficient to download the data at the start of the application and call the service if necessary in case of significant location changes.

5. Conclusion

In this work, the focus was on the analysis of resource consumption in a sensor-based, mobile AR application, SARAS. As the resources, it uses motion sensors and the graphics API in addition to GPS, the camera and the web services for the content of bank data. Firstly, CPU profiling using the methods available in Android platforms were examined and improvements suggested by these methods are applied to SARAS. The functions and methods that consume high power were determined and modified. These improvements reduced the resource consumptions up to %35. Afterwards, a detailed energy analysis was performed under different GPS and sensor processsing settings. Finally, the comparison with other applications were performed in terms of energy consumption. As a result, it has been observed that SARAS consumes similar amount of energy compared to similar commercial applications like LAYAR.

6. Acknowledgments

This work is supported Galatasaray University Research Fund under the grant numbers 15.401.004 and 15.401.006, and by the SAN-TEZ program of Turkish Ministry of Science, Industry and Technology under the grant number 0307.STZ.2013-2.

References

1. Zhuang, Z., Kim, K.H., Singh, J.P.. Improving energy efficiency of location sensing on smartphones. In: *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*; MobiSys '10. ACM. ISBN 978-1-60558-985-5; 2010, p. 315–330.
2. Sarmiento, A.L., Amor, M., Padrón, E.J., Regueiro, C.V., Concheiro, R., Quintıa, P.. Evaluating performance of android systems as a platform for augmented reality applications. *International Journal on Advances in Software Volume 5, Number 3 & 4, 2012* 2012;.
3. Wagner, D., Schmalstieg, D.. Making augmented reality practical on mobile phones, part 1. *Computer Graphics and Applications, IEEE* 2009;**29**(3):12–15. doi:10.1109/MCG.2009.46.
4. Chen, M., Ling, C., Zhang, W.. Analysis of augmented reality application based on cloud computing. In: *Image and Signal Processing (CISP), 2011 4th International Congress on*; vol. 2. IEEE; 2011, p. 569–572.
5. Layar application. last accessed, January 2016. URL: `https://www.layar.com/`.
6. Wikitude app. last accessed, January 2016. URL: `http://www.wikitude.com/`.
7. Zhang, L., Tiwana, B., Qian, Z., Wang, Z., Dick, R.P., Mao, Z.M., et al. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In: *Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*; CODES/ISSS '10. ACM. ISBN 978-1-60558-905-3; 2010, p. 105–114.
8. Sood, R.. *Pro Android augmented reality*. Apress; 2012. ISBN 9781430239451.